LOW POWER AND AREA OPTIMIZED APPROXIMATE BOOTH MULTIPLIERS WITH ERROR COMPENSATION AND COMPRESSION

R.Gowri Shankar[1], D.R. Ananthi,[2]
Professor & Head[1], M.E (VLSI Design)[2]
[1,2]Department of Electronics and Communication Engineering,
[1,2]KIT-Kalaignarkarunanidhi institute of technology, Coimbatore-641402
rgowri2017@gmail.com[1], ananthidevaraj1312@gmail.com[2]

Abstract: Approximate computing is a promising technique to elevate the performance of digital circuits which curtail the hardware requirements by exploiting the inherent error resilience of certain applications such as digital signal processing, multimedia and machine learning. Approximate multipliers and other approximation techniques can be integrated to increase the performance of applications such as convolutional Neural Networks. In the proposed work, a truncation based Booth multiplier is designed based on multi-level compressors such as 4:2, 5:2 and 6:3 counter. A compensation circuit is generated by selective modifications in k-map to circumvent the carry appearing from the truncated part. By efficient mapping, hardware pruning and output error reduction is achieved simultaneously. In the quest of power and accuracy trade-off, Truncated and Approximate Booth Multipliers using compressors and counters (TABM-CC) are proposed with a range of designs based on truncation factor w. When compared with the state-of-the-art multipliers, TACBM outperforms in terms of accuracy and Area-Power savings.

Key words: Truncated and Approximate Booth Multipliers using compressors and counters (TABM-CC), Compressors, Counters, Digital Signal Processing

Corresponding Author: *R. Gowri Shankar*
Department of ECE
KIT- Kalaignarkarunanidhi Institute of Technology
Mail: rgowri2017@gmail.com

Introduction:

Because of their higher efficiency, parallel multipliers are now extremely used in many high-speed systems such as digital signal processors (DSPs), central processing units (CPUs), and multimedia applications [1]. In most of the CPUs, the multiplier lies in the critical path for signal propagation. To decrease the delay and complexity of such systems, practical design considerations have been pursued over recent years [2]. In a parallel multiplier, the multiplication process is divided into three steps. At first, the partial products (PPs) are generated. Then these products are summed, and the process continues until two rows remain, and at the final stage, the two remaining rows will be added by means of, for example, a carry propagation adder [3]. At the second stage and after generation of PPs, a partial product reduction tree (PPRT) is often employed for efficient summation of the products. Considering the full adder (FA) as the main building block in various multiplication configurations [4], this block constitutes the basis for much different architecture.

However, the main drawback of FA-based configurations, which limits their usage in today's parallel multiplier design, is the propagation latency for the cascaded cells. Moreover, the most important concern in the design process of a parallel multiplier is the circuit size and power consumption, which is directly related to the number of employed gates used in the various parts of the architecture [5]. An efficient solution to overcome such drawbacks is to utilize a compressor network instead of FA trees, especially in the PPRT. Furthermore, comprehensive analysis depicts that the most significant part of the total delay and power dissipation will belong to this stage in a parallel multiplier. Therefore, the performance enhancement of this stage can significantly improve the speed and lower the power dissipations of the whole system [3], [6].

As a universal principle, an $n - 2$ compressor modifies the interconnection of the adjacent cells in the accumulation stage by means of one or more horizontal paths defined as *C*out1, *C*out2, etc. [7], [8]. These paths will transmit their logic value to the adjoining compressor, which has one binary bit higher order in significance known as *C*in1, *C*in2, etc. for the latter compressor, respectively. By reducing the signal transfer load for the vertical trajectories known as Sum and Carry outputs, significant performance enhancement can be achieved [9]. However, the carry rippling will be unfolded as a new problem for such structures that must be taken into account. Such an obstacle has barely been considered in previously reported works. Considering the

above, the focus in this article is to improve the speed performance of $n - 2$ compressors by bringing up the carry rippling problem as a new design consideration. Also, the fact that power and active area on-chip should be comparable to the best-reported designs is focused. Therefore, the main emphasis was on the neutral states of the outputs for horizontal paths. Utilization of the newly proposed truth table has resulted in a fast 303-ps 5-2 compressor (designed in a CMOS standard 0.18-μm process) in which the gate-level delay is fewer than three XOR logic gates. By expanding the same idea for the 7-2 compressor, the gate-level latency of four XOR logic gates is obtained, which is a considerable speed optimization in compressor design criteria.

Multipliers are essential components of digital hardware, ranging from deeply embedded system on- chip (SoC) cores to GPU-based accelerators. As they are often critical for system performance, a great emphasis was placed on their performance improvement in the past few decades [1]–[7]. While performance remains important, the high demand for battery-powered ubiquitous systems has promoted low-power operation to a primary design goal [8].. Current implementations of binary multiplication follow the steps of [7]: 1) recoding of the multiplier in digits in a certain number system; 2) digit multiplication of each digit by the multiplicand, resulting in a certain number of partial products; 3) reduction of the partial product array to two operands using multi operand addition techniques; and 4) carry-propagate addition of the two operands to obtain the final result.

LITERATURE REVIEW

1)An Energy-Efficient Multiplier With Fully Overlapped Partial Products Reduction and Final Addition

An energy-efficient fast array multiplier is proposed and designed. The multiplier operates in a left-to-right mode enabling a full overlap between reduction of partial products in carry-save form and the final addition producing the product. The design is based on the left-to-right carry-free (LRCF) multiplier. It differs from the LRCF multiplier in a much smaller on-the-fly conversion circuit of O(n) size and the use of radix-4 full adders in the conversion. The new converter produces the most-significant half of the product during the reduction process. It eliminates the most-significant part of the final adder. The least-significant half of the product is obtained with

a carry-ripple adder during the reduction. Thus conversion of the carry-save form of accumulated partial products to the conventional product does not add any delay to the total time of the multiplier. Several right-to-left, left to- right multipliers and tree multipliers are designed for 16, 24, 32, and 56 bits, and radices 2 and 4, synthesized in 90 nm technology and compared, demonstrating the advantages and disadvantages of the proposed design with respect to area, delay, power, and energy. We considered both truncated and full-precision multipliers. The proposed multiplier has lower delay, area, power, and energy than other considered types of array multipliers. Its advantages grow with the increase in precision. As expected, it is slower than a tree multiplier but it has smaller area, power, and energy.

2)Design and Implementation of High-Speed and Energy-Efficient Variable-Latency Speculating Booth Multiplier (VLSBM)

Data hazards cause severe pipeline performance degradation for data-intensive computing processes. To improve the performance under a pessimistic assumption on the pipeline efficiency, a high-speed and energy-efficient VLSBM is proposed that successively performs a speculating and correcting phase. To reduce the critical path, the VLSBM partial products are partitioned into the (n-z)-bit least significant part (LSP) and the self-reliant (n+z)-bit most significant part (MSP), and an estimation function stochastically predicts the carry to the MSP, thereby allowing independent calculation of the partial-product accumulation of parts. When a carry prediction is accurate, the data dependence is hidden and the correcting phase is bypassed, thereby ensuring the potential speed-up of the pipelined datapath. If a prediction is inaccurate, the speculation is flushed and the correcting phase is executed to obtain the exact multiplication. The simulation results verify the effectiveness of the proposed VLSBM. When applied to a DSP algorithm with a data hazard (or dependence) probability ,Pd, $0<Pd<1$, the results show that the proposed VLSBM outperforms the original Booth multiplier and the fastest conventional well-pipelined modified Booth multiplier when $Pd>0.32$. For the case of high Pd with Pd=1, the proposed VLSBM improves approximately 1.47 times speedup against the fastest conventional pipelined Booth multiplier (@UMC 90 nm CMOS) and, furthermore, approximately 25.4% of energy per multiplication and 7% of area are saved. By examining multiplications during three multimedia application processes (i.e., JPEG compression, object detection, and H.264/AVC

decoding), the proposed VLSBM improves the speed-up ratio by approximately 1.0 to 1.4 times, and reduces the cycle count ratio by approximately 1.3 to 1.8 times in comparison to the fastest conventional two-stage pipelined Booth multiplier.

3) An Accuracy-Adjustment Fixed-Width Booth Multiplier Based on Multilevel Conditional Probability

This brief proposes an accuracy-adjustment fixed width Booth multiplier that compensates the truncation error using a multilevel conditional probability (MLCP) estimator and derives a closed form for various bit widths L and column information w. Compared with the exhaustive simulations strategy, the proposed MLCP estimator substantially reduces simulation time and easily adjusts accuracy based on mathematical derivations. Unlike previous conditional-probability methods, the proposed MLCP uses entire nonzero code, namely MLCP, to estimate the truncation error and achieve higher accuracy levels. Furthermore, the simple and small MLCP compensated circuit is proposed in this brief. The results of this brief show that the proposed MLCP Booth multipliers achieve low-cost high accuracy performance.

4) A Modified Partial Product Generator for Redundant Binary Multipliers

Due to its high modularity and carry-free addition, a redundant binary (RB) representation can be used when designing high performance multipliers. The conventional RB multiplier requires an additional RB partial product (RBPP) row, because an error-correcting word (ECW) is generated by both the radix-4 Modified Booth encoding (MBE) and the RB encoding. This incurs in an additional RBPP accumulation stage for the MBE multiplier. In this paper, a new RB modified partial product generator (RBMPPG) is proposed; it removes the extra ECW and hence, it saves one RBPP accumulation stage. Therefore, the proposed RBMPPG generates fewer partial product rows than a conventional RB MBE multiplier. Simulation results show that the proposed RBMPPG based designs significantly improve the area and power consumption when the word length of each operand in the multiplier is at least 32 bits; these reductions over previous NB multiplier designs incur in a modest delay increase (approximately 5%). The power-delay product can be reduced by up to 59% using the proposed RB multipliers when compared with existing RB multipliers.

5)Approximate Radix-8 Booth Multipliers for Low-Power and High-Performance Operation

The Booth multiplier has been widely used for high performance signed multiplication by encoding and thereby reducing the number of partial products. A multiplier using the radix-4 (or modified Booth) algorithm is very efficient due to the ease of partial product generation, whereas the radix-8 Booth multiplier is slow due to the complexity of generating the odd multiples of the multiplicand. In this paper, this issue is alleviated by the application of approximate designs. An approximate 2-bit adder is deliberately designed for calculating the sum of 1X and 2X of a binary number. This adder requires a small area, a low power and a short critical path delay. Subsequently, the 2-bit adder is employed to implement the less significant section of a recoding adder for generating the triple multiplicand with no carry propagation. In the pursuit of a trade-off between accuracy and power consumption, two signed 16X16 bit approximate radix-8 Booth multipliers are designed using the approximate recoding adder with and without the truncation of a number of less significant bits in the partial products. The proposed approximate multipliers are faster and more power efficient than the accurate Booth multiplier; moreover, the multiplier with 15-bit truncation achieves the best overall performance in terms of hardware and accuracy when compared to other approximate Booth multiplier designs. Finally, the approximate multipliers are applied to the design of a low-pass FIR filter and they show better performance than other approximate Booth multipliers.

METHODOLOGY

MULTIPLIER

For digital signal processing and applications involving signal and image processing, multipliers and adders form an important part. So, speed of the multipliers and adders affect speed of the operation. For high speed applications, faster multipliers are recommended. Several techniques to increase multiplier speed are proposed. Multiplication is done by adding partial product terms. Implementation of multiplier comprises three steps:

- generation of partial products
- partial products reduction tree

- a vector merge addition to produce final product from the sum and carry rows generated from the reduction tree.

Second step consumes more power. In this approximation is applied in reduction tree stage.

Partial product accumulation in 4 × 4 multipliers can be represented as, consider two 4-bit unsigned operands $\alpha = \sum_{i=0}^{3} \alpha_i 2^i$ and $\beta = \sum_{j=0}^{3} \beta_j 2^j$. The partial product array pp is a 4×4-bit array of the partial product bits $pp_{i,j} = \alpha_i \beta_j$, where $i, j \in \{0, 1, 2, 3\}$. Table 3.1 gives all the partial products for a 4-bit multiplication and their corresponding product bits.

Table 3.1 Original partial product of the multiplication

Stage 7	Stage 6	Stage 5	Stage 4	Stage 3	Stage 2	Stage 1	Stage 0
	$pp_{3,3}$	$pp_{3,2}$	$pp_{3,1}$	$pp_{3,0}$	$pp_{2,0}$	$pp_{1,0}$	$pp_{0,0}$
		$pp_{2,3}$	$pp_{2,2}$	$pp_{2,1}$	$pp_{1,1}$	$pp_{0,1}$	
			$pp_{1,3}$	$pp_{1,2}$	$pp_{0,2}$		
				$pp_{0,3}$			
γ_7	γ_6	γ_5	γ_4	γ_3	γ_2	γ_1	γ_0

The product is denoted by $\gamma = \sum_{k=0}^{7} \gamma_k 2^k$. The bits of γ are produced in stages going from the LSB to the MSB. According to Table 1.1, $\gamma_0 = pp_{0,0}$ and there is no further operation in Stage 0. In Stage 1, to generate γ_1, we can simply use a half adder that produces a sum bit γ_1 and a carry bit (c_1) for the next stage. Since the half adder circuit is already a simple design, there is no need to approximate it.In Stage 2, there are three pp terms and the carry from the previous stage (c_1) that must be added together. Thus, a 4:2 compressor is required to generate γ_1 and a carry for the next stage.

NOVEL CIRCUITS FOR MBE

As observed in earlier research, a proper choice of intermediate signals in the interface between Booth encoding and decoding offers opportunities for logic optimization. Fig. 3.1(a)–(d) illustrates the traditional implementations of MBE circuits found in the literature. Note that only the full-swing circuit topologies were considered in this study. Fig. 3.1(a) (BED13) depicts a hybrid implementation of encoder–decoder circuits which require 36 and 10 transistors, respectively. This non-CMOS implementation reports the least number of transistors for the decoder block among the presented.

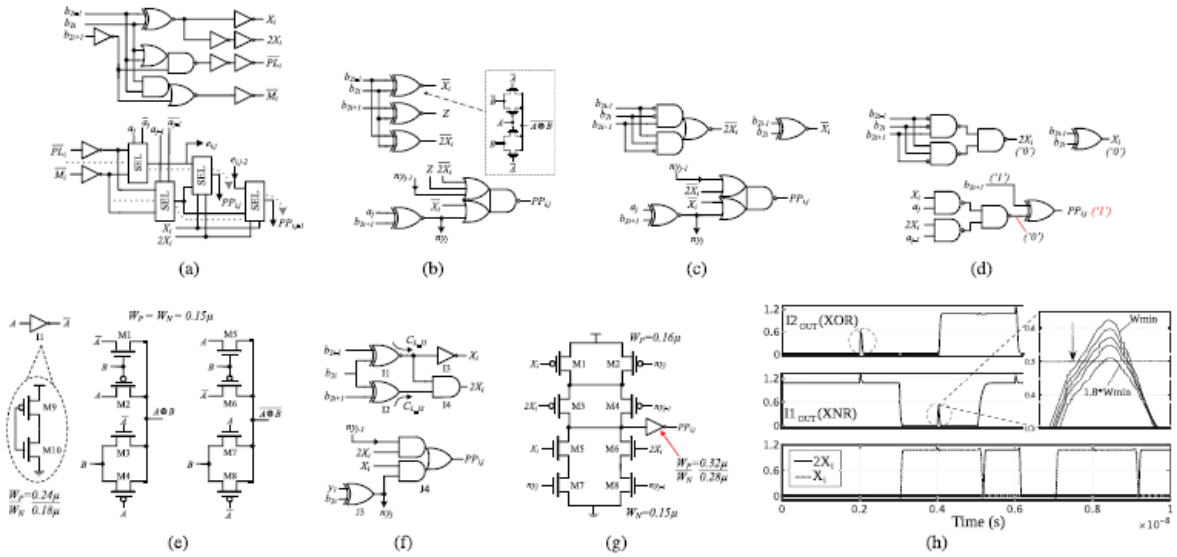


Fig. 3.1. Various Booth encoder–decoder implementations. (a) BED13 [46]. (b) BED20 [27]. (c) BED22 [7], [16], [41]. (d) Erroneous Booth circuits in [17]. (e) 6T-XOR/XNR circuits of this work (WM1–M8 = 0.15µ). (f) Proposed encoder–decoder circuits (BED18). (g) AO22 (J3) of the decoder (WM1–M4 =0.16 µ, WM5–M8 = 0.15 µ).

However, there are a few issues that emanate from this implementation. First, the unbuffered selector circuit which is denoted by SEL (composed of four pass transistors), forms cascaded resistive paths from decoder inputs to the outputs as highlighted in Fig. 3.1(a). This results in an asymmetry in the driving loads to the SEL blocks for different input combinations and therefore

different arrival times. Secondly, the routing congestion across the decoding blocks in Fig. 3.1(a) is relatively higher and increases the interconnect parasitics across the PPG.

The circuits shown in Fig. 3.1(b) (BED20) [27] uses transmission gate pairs for encoders leading to a faster operation in PPG. However the unbuffered encoder outputs become transparent to the hazards induced by the circuit itself. The additional wiring and higher capacitive loading at the decoder leads to a higher power consumption in PPG at the same time. The arrangement in Fig. 3.1(c) (BED22) [7], [16] is the most optimized version in terms of transistor count and signal synchronization. The XORs which produce ny j−1–ny j are shared among the decoders and the AOI22 cell provides balanced loads to the encoder signals. Therefore, it was also preferred for the truncated multiplication in [41]. The unique Booth circuits presented in [17] and [44] are not considered for the evaluation due to functional failures when all the encoder inputs (b2i−1–b2i+1) are at logic "1" [see Fig. 3.1(d)]. The proposed MBE circuits in this work are shown in Fig. 3.1(e)–(g).

The essential leaf cell of the proposed circuitry is depicted in Fig. 3.1(e). This XOR/XNR arrangement results in fewer number of gate capacitances when compared to any other full-swing implementations [47]–[49]. Despite this merit, it suffers from the delay asymmetry between the signal paths. If, for example, in the circuit of Fig. 3.1(e), when both inputs change from 0_1, M1 of the XOR drives the output for a short period of time due to the inertial and propagation delays of the inverter and as a result, a glitch appears at the XOR output. The inversely proportional relationship between the inertial and propagation delays limits the liberty of device sizing. As such, the direct interfacing of these XOR/XNR outputs to high fan-out nets could only worsen the spurious activities in PPG.

Fig.3.5 shows the proposed block diagram which consists of partial product generation and partial product reduction using compressors. 4:2 and 5:2 compressors are used for reducing the complexity in partial product reduction

PROPOSED BLOCK DIAGRAM

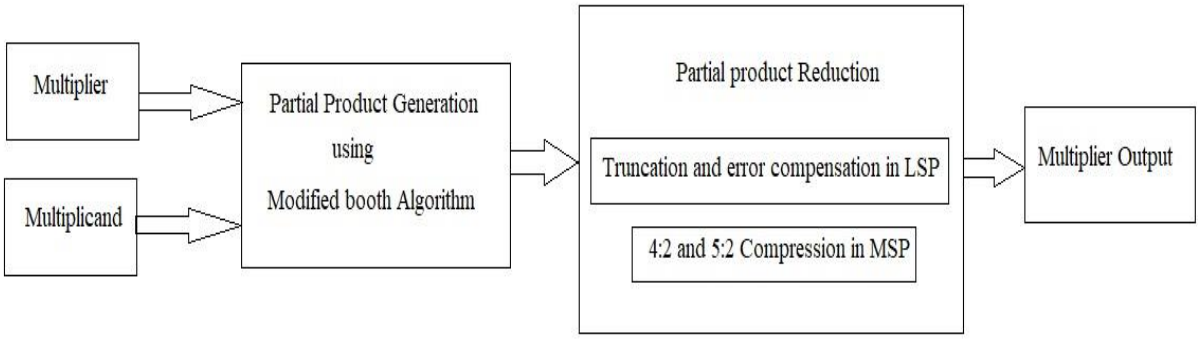


Fig.3.2 Proposed block diagram

EXACT 4:2 COMPRESSOR

The exact compressors have *5* inputs x_1, x_2, x_3, x_4 and c_{in}. It generates *3* outputs Carry, c_{out} and Sum. Compressor reduces n number of bits to 2 bits by proper repetition. Carry bit c_{out} goes to higher position.The function of the exact 4:2 compressor is implemented by using two appropriately connected full adders Fig.3.6 as given by

$$Sum = x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus C_{in},$$

$$C_{out} = (x_1 \oplus x_2).x_3 + \overline{(x_1 \oplus x_2)}.x_1, \quad (1)$$

$$Carry = (x_1 \oplus x_2 \oplus x_3 \oplus x_4).C_{in} + \overline{(x_1 \oplus x_2 \oplus x_3 \oplus x_4)}.x_4.$$

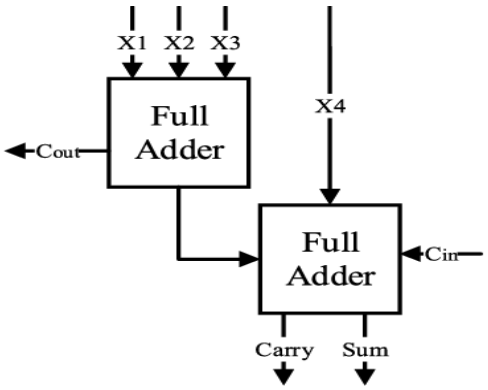


Fig.3.3 Exact Compressor Design

The sum output has the same weight as the four input signals while thec_{out} is used as the carry in for the next higher-order compressor and the output Carry is weighted like a pp bit in a one-bit-higher position. Note that c_{out} and Carry have the same weight. The two stages of an exact 4:2 compressor chain are shown in Fig.3.7.

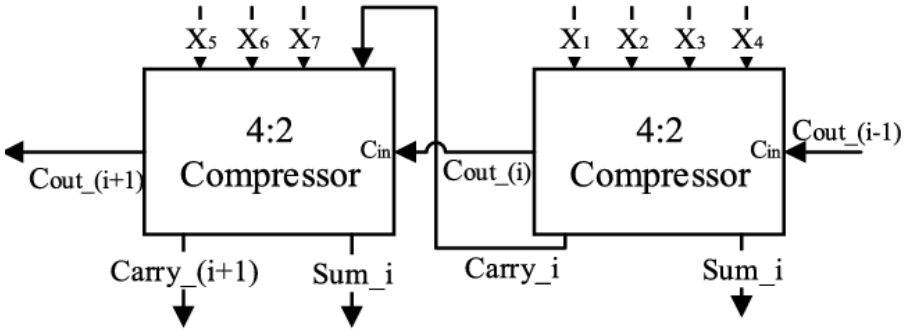


Fig.3.4 Exact Compressor chain

5-2 COMPRESSOR

To design the new circuit, some changes are applied to the conventional truth table of 5-2 compressor. The new truth table is illustrated in Fig. 3.8. According to the logic state of Cin1 and Cin2, there will be three separate tables. The hatched regions divide each table into four parts to cover all possible combinational states of I1, I2, and I3 inputs.

Cout1 and Cout2 are planned to be independent of Cin1 and Cin2 for all states to enhance the speed performance in a simpler scheme. It also prevents any carry rippling when Cout1 and Cout2 are being produced because Cin1 and Cin2 come from neighbor compressor cells. Based on the truth table of Fig. 3.8, Cout1 always rises to high-level voltage, if at least two of input signals have logic "1" value. Therefore, we have

Cout1 = I1 · I2 + I1 · I3 + I2 · I3. (2)

Also, Cout2 can be obtained using the following equation:

Cout2 =I4 or I5 I4 = I5

I1 xor I2 xor I3 I4 /= I5.(3)

$C_{in1}=0$, $C_{in2}=0$

	n*	C_{out2}	C_{out1}	Carry	Sum		C_{out2}	C_{out1}	Carry	Sum	
$I_4=0$ $I_5=0$	0	0	0	0	0		0	0	0	1	$I_4=0$ $I_5=1$
	1	0	0	0	1		1	0	0	0	
	2	0	1	0	0		0	1	0	1	
	3	0	1	0	1		1	1	0	0	
$I_4=1$ $I_5=0$	0	0	0	0	1		1	0	0	0	$I_4=1$ $I_5=1$
	1	1	0	0	0		1	0	0	1	
	2	0	1	0	1		1	1	0	0	
	3	1	1	0	0		1	1	0	1	

$C_{in1}=0$, $C_{in2}=1$ / $C_{in1}=1$, $C_{in2}=0$

	n*	C_{out2}	C_{out1}	Carry	Sum		C_{out2}	C_{out1}	Carry	Sum	
$I_4=0$ $I_5=0$	0	0	0	0	1		0	0	1	0	$I_4=0$ $I_5=1$
	1	0	0	1	0		0	0	0	1	
	2	0	1	0	1		0	1	1	0	
	3	0	1	1	0		0	1	0	1	
$I_4=1$ $I_5=0$	0	0	0	1	0		1	0	0	1	$I_4=1$ $I_5=1$
	1	1	0	0	1		1	0	1	0	
	2	0	1	1	0		1	1	0	1	
	3	1	1	0	1		1	1	1	0	

$C_{in1}=1$, $C_{in2}=1$

	n*	C_{out2}	C_{out1}	Carry	Sum		C_{out2}	C_{out1}	Carry	Sum	
$I_4=0$ $I_5=0$	0	0	0	1	0		0	0	1	1	$I_4=0$ $I_5=1$
	1	0	0	1	1		1	0	1	0	
	2	0	1	1	0		0	1	1	1	
	3	0	1	1	1		1	1	1	0	
$I_4=1$ $I_5=0$	0	0	0	1	1		1	0	1	0	$I_4=1$ $I_5=1$
	1	1	0	1	0		1	0	1	1	
	2	0	1	1	1		1	1	1	0	
	3	1	1	1	0		1	1	1	1	

* n is number of I_1 , I_2 and I_3 inputs in which they are equal to logic 1.

Fig. 3.5. Proposed truth table for implementation of the 5-2 compressor.

According to (21), the Sum output attains a high-level voltage only if an odd number of inputs have logic "1" value

Sum = I1 xor I2 xor I3 xor I4 xor I5 xor Cin1 xor Cin2. (4)

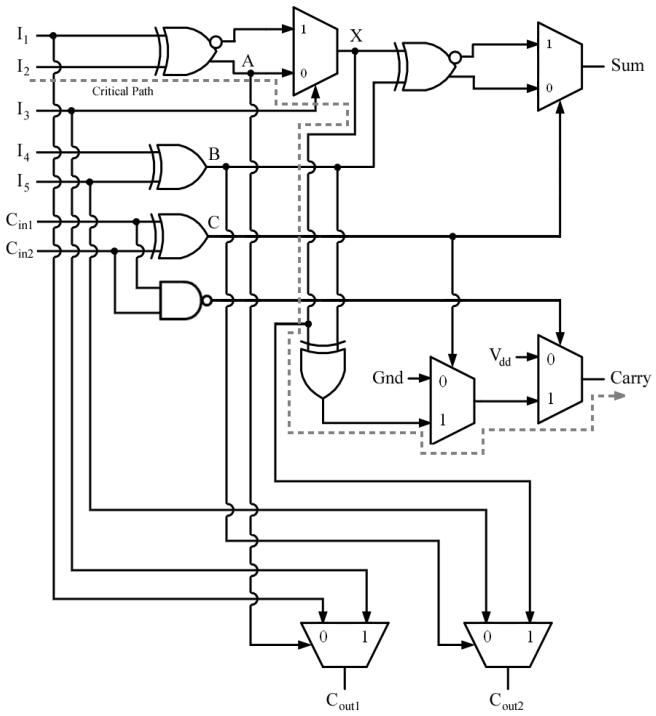


Fig. 3.6. Proposed 5-2 compressor.

The newly proposed circuit for 5-2 compressor has been demonstrated in Fig. 3.9. For the better realization of the latency in different paths from inputs to the outputs, two neighbor 5-2 compressor cells must be connected together as the carry rippling finishes in the second compressor block. This point is discussed in the simulation section. By considering this fact and as shown in Fig. 3.9, the critical path will belong to the generation of Carry output. It must be taken into account that all of the MUX gates in Fig. 3.9 except the one that produces Cout1, and the gate which is fed by Gnd, are channel-ready gates. As a result, they will exhibit a latency equal to 0.25 delta. Moreover, Bbar is produced by means of an inverter at the output node of the nonfull swing XOR gate that generates B. However, it does not affect the delay of the whole system. For high logic values at the input stage of TG, the output capacitor will be charged to Vdd–Vth by means of the NMOS transistor. At the same time, Bbar is transferred to the output of the inverter gate. With the help of Bbar, which enables the PMOS transistor, the output

capacitor will fully be charged to Vdd value. For low logic states in the input stage of TG, the NMOS path itself discharges the output capacitor to the zero value.

The two output XOR–XNOR gates generate both outputs simultaneously, which are designed and wholly discussed in [8]. Because of the similar paths, their different outputs have little influence on the delay and glitch effect of TG waveforms. Also, static CMOS has been employed to design the architectures of NAND and NOR gates. If the outputs of these gates are inverted, then the AND/OR gates will be obtained, and since these gates have not been located in the critical path, therefore, they would not affect the latency of the whole system. To calculate the delay of the critical path for the proposed 5-2 compressor in the logic gate level, we refer to the calculations provided in the previous section. As stated above, the critical path belongs to the route starting from I1 and I2 and finishing in Carry output.

Results

MODELSIM

ModelSim is a multi-language HDL simulation environment by Mentor Graphics, for simulation of hardware description languages such as VHDL, Verilog and SystemC, and includes a built-in C debugger. ModelSim can be used independently, or in conjunction with Altera Quartus or XilinxISE. Simulation is performed using the graphical user interface (GUI), or automatically using scripts. The ModelSim environment is shown in Fig.4.2.ModelSim is offered in multiple editions, such as ModelSim PE, ModelSimSE, and ModelSim XE. ModelSim SE offers high-performance and advanced debugging capabilities, while ModelSim PE is the entry-level simulator for hobbyists and students. ModelSim SE is used in large multi-million gate designs, and is supported on Microsoft Windows and Linux, in 32-bit and 64-bit architectures.

ModelSim XE stands for Xilinx Edition, and is specially designed for integration with Xilinx ISE. ModelSim XE enables testing of HDL programs written for Xilinx Virtex/Spartan series FPGA's without needed physical hardware.

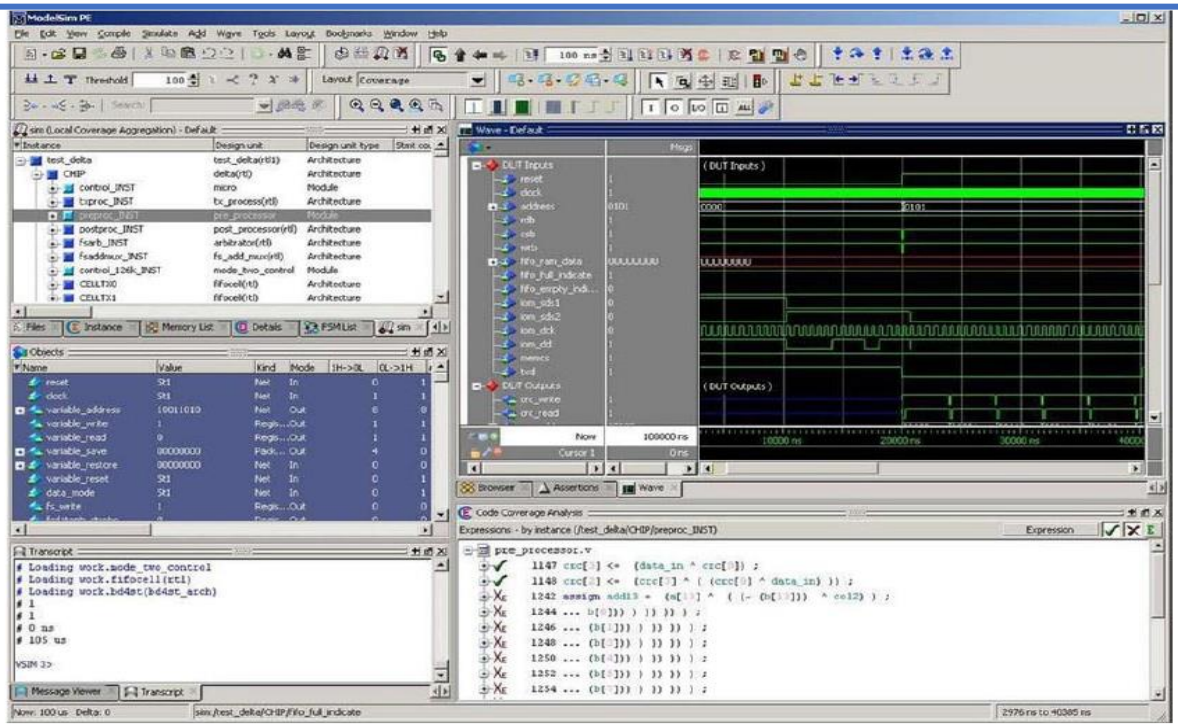


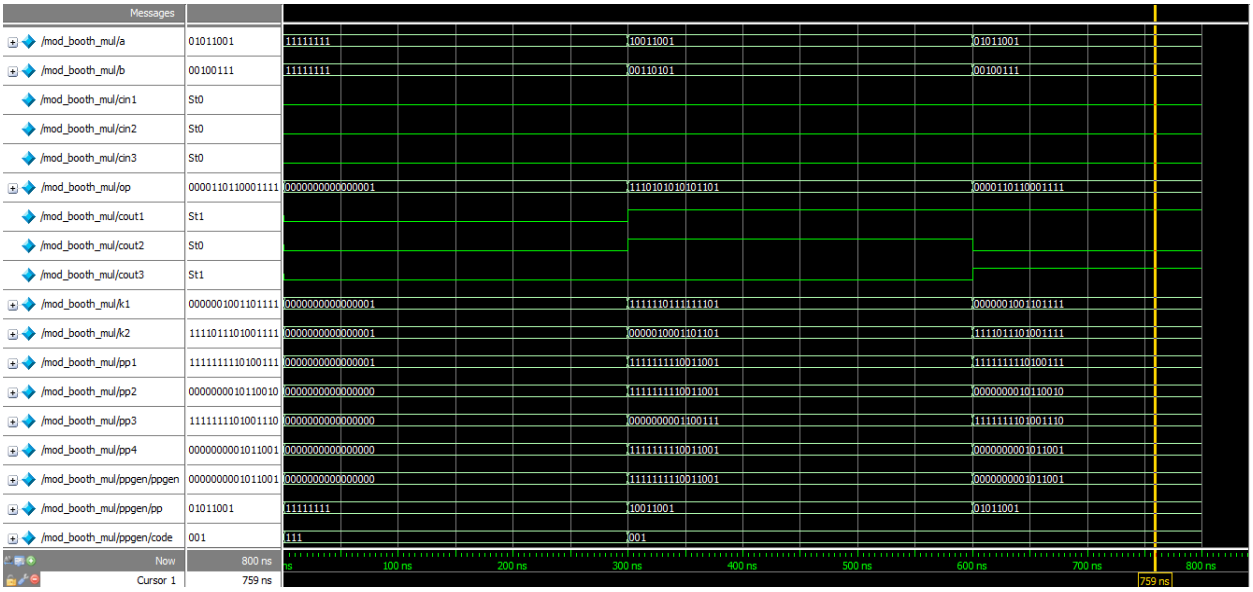
Fig.4.1.ModelSim Environment

ModelSim can also be used with MATLAB/Simulink, using Link for ModelSim. Link for ModelSim is a fast bidirectional co-simulation interface between Simulink and ModelSim. For such designs, MATLAB provides a numerical simulation toolset, while ModelSim provides tools to verify the hardware implementation & timing characteristics of the design.

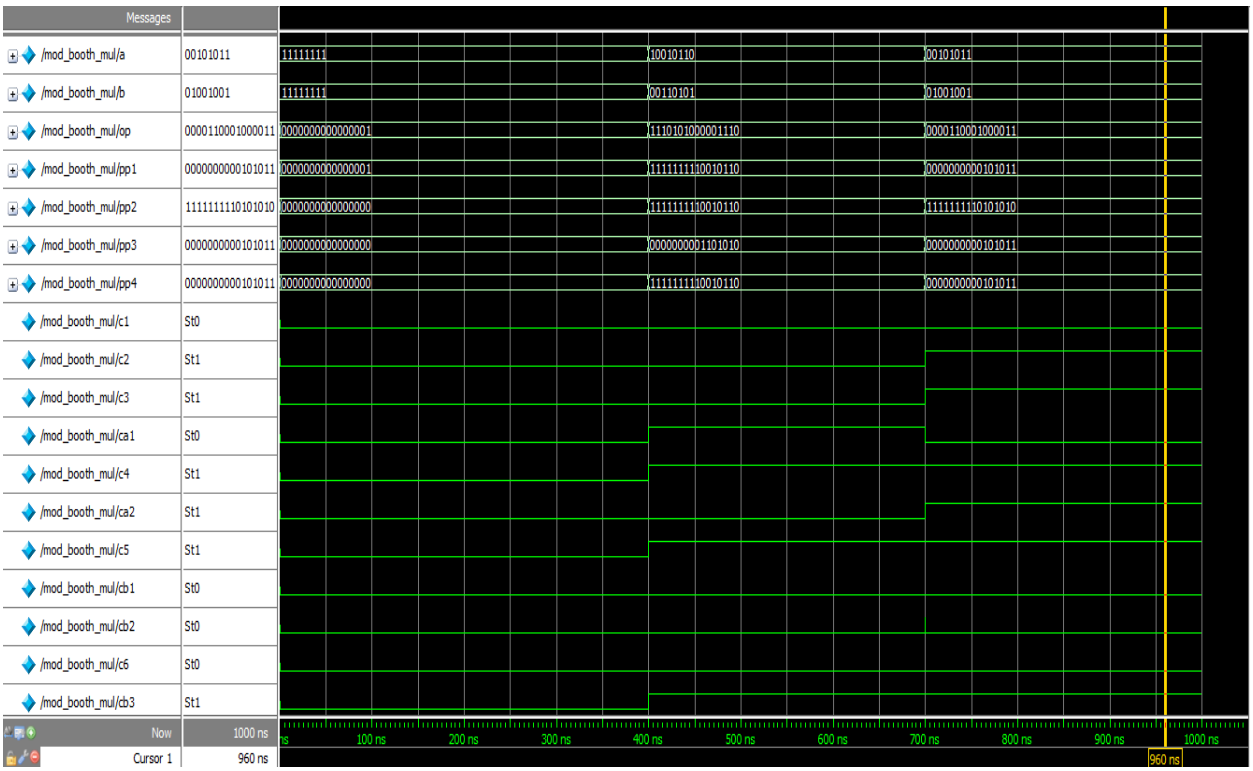
ModelSim eases the process of finding design defects with an intelligently engineered debug environment. The ModelSim debug environment efficiently displays design data for analysis and debug of all languages.

ModelSim allows many debug and analysis capabilities to be employed post-simulation on saved results, as well as during live simulation runs.

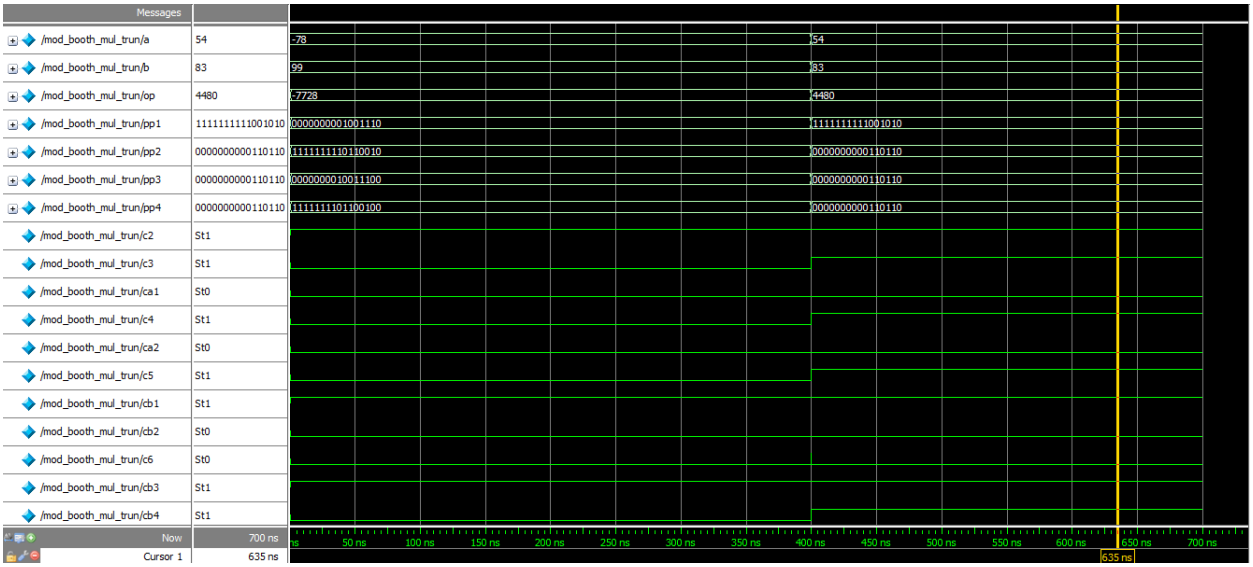
SIMULATION RESULT OF EXISTING MULTIPLIER



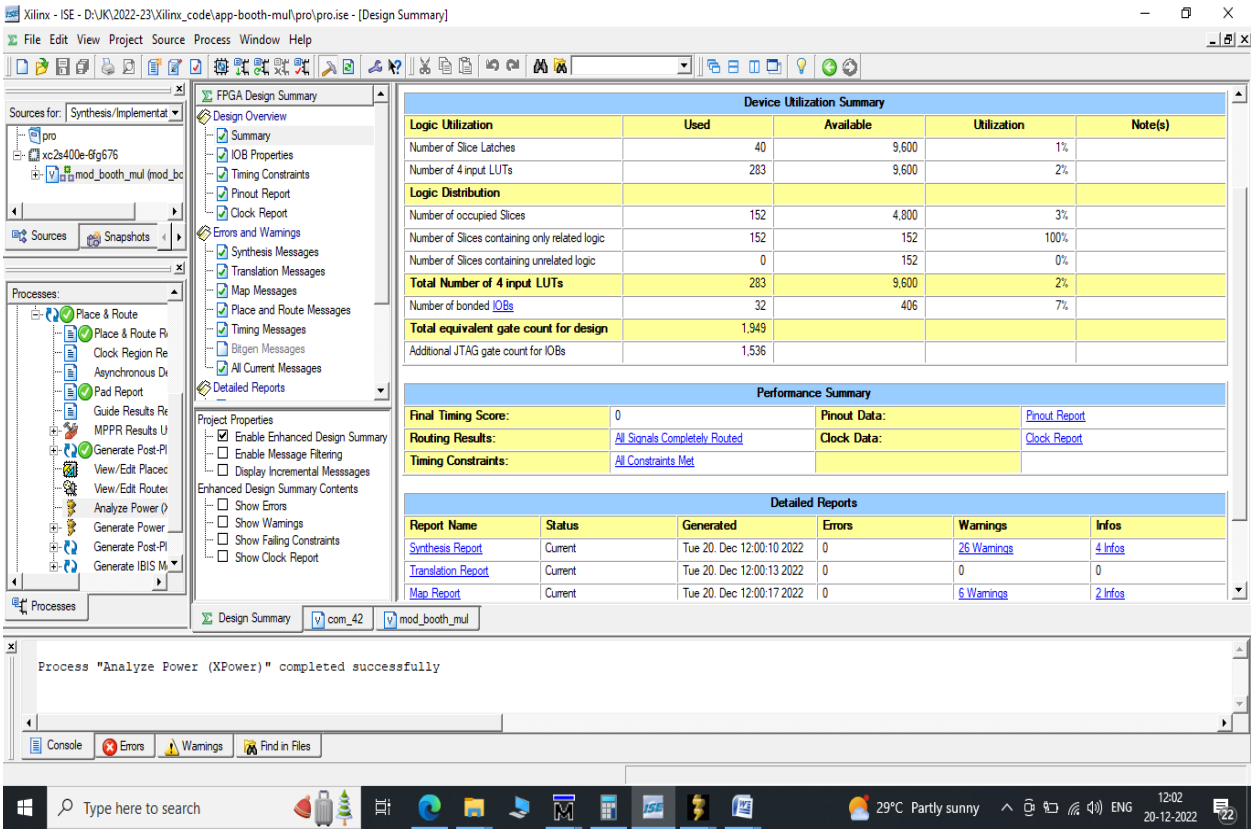
SIMULATION RESULT OF PROPOSED MULTIPLIER



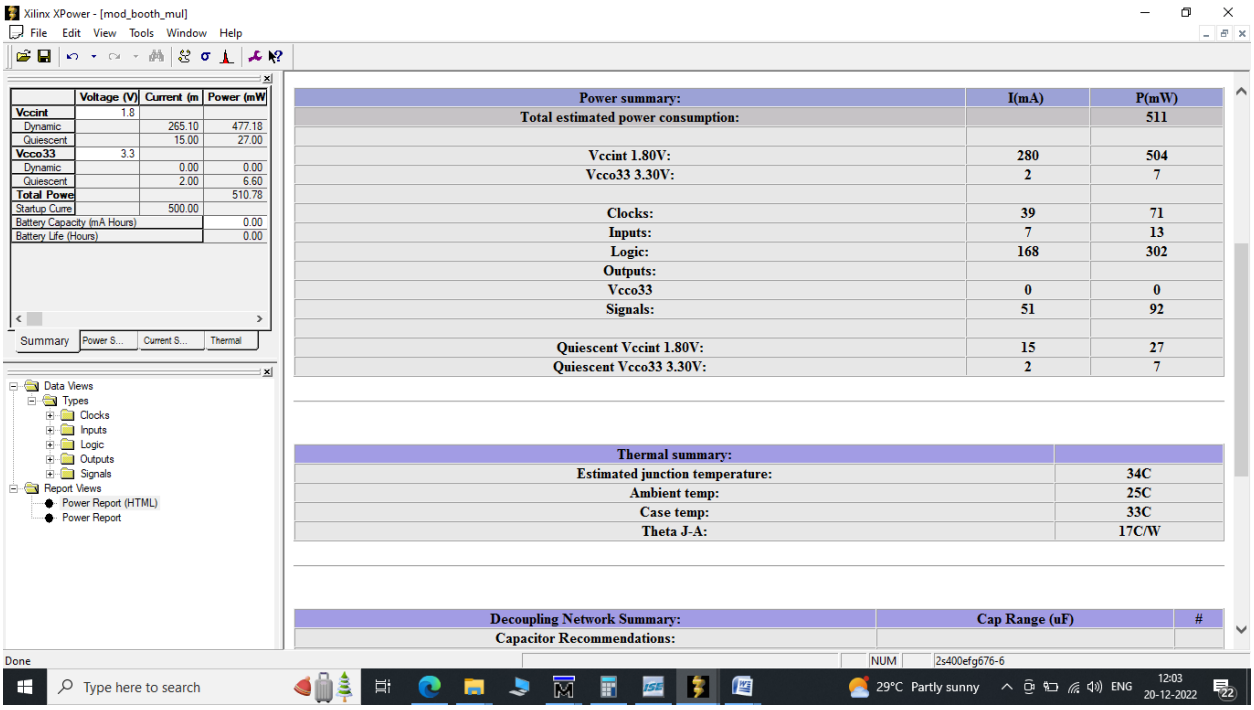
SIMULATION RESULT OF PROPOSED TRUNCATED BOOTH MULTIPLIER



AREA REPORT OF PROPOSED METHOD



POWER REPORT OF PROPOSED METHOD



Xilinx XPower - [mod_booth_mul]

File Edit View Tools Window Help

	Voltage (V)	Current (m	Power (mW
Vccint	1.8		
Dynamic		265.10	477.18
Quiescent		15.00	27.00
Vcco33	3.3		
Dynamic		0.00	0.00
Quiescent		2.00	6.60
Total Powe			510.78
Startup Curre		500.00	
Battery Capacity (mA Hours)			0.00
Battery Life (Hours)			0.00

Summary | Power S... | Current S... | Thermal

Data Views
- Types
  - Clocks
  - Inputs
  - Logic
  - Outputs
  - Signals
- Report Views
  - Power Report (HTML)
  - Power Report

Power summary:	I(mA)	P(mW)
Total estimated power consumption:		511
Vccint 1.80V:	280	504
Vcco33 3.30V:	2	7
Clocks:	39	71
Inputs:	7	13
Logic:	168	302
Outputs:		
Vcco33	0	0
Signals:	51	92
Quiescent Vccint 1.80V:	15	27
Quiescent Vcco33 3.30V:	2	7

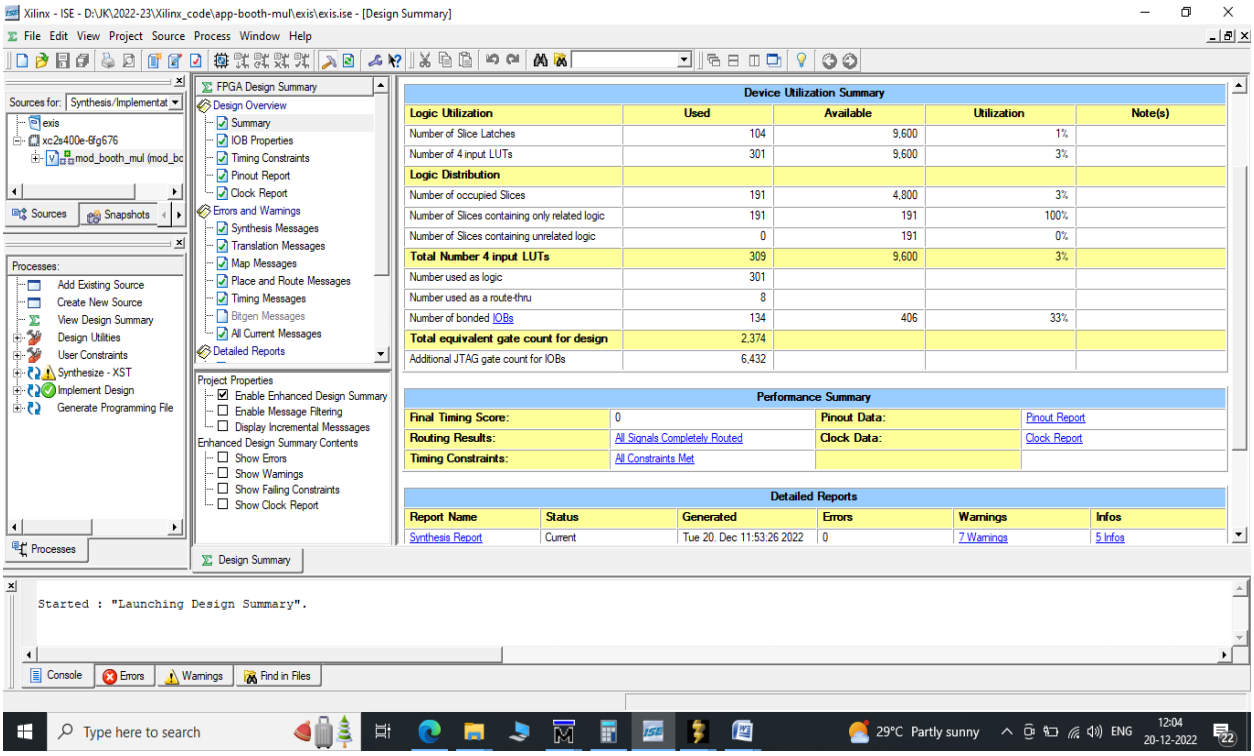
Thermal summary:	
Estimated junction temperature:	34C
Ambient temp:	25C
Case temp:	33C
Theta J-A:	17C/W

Decoupling Network Summary:	Cap Range (uF)	#
Capacitor Recommendations:		

Done NUM 2s400efg676-6

Type here to search 29°C Partly sunny ENG 12:03 20-12-2022

AREA REPORT OF EXISTING METHOD



Xilinx - ISE - D:\JK\2022-23\Xilinx_code\app-booth-mul\exis\exis.ise - [Design Summary]

File Edit View Project Source Process Window Help

Sources for: Synthesis/Implementat
- exis
- xc2s400e-6fg676
- mod_booth_mul (mod_bo

Sources | Snapshots

Processes:
- Add Existing Source
- Create New Source
- View Design Summary
- Design Utilities
- User Constraints
- Synthesize - XST
- Implement Design
- Generate Programming File

Processes

FPGA Design Summary
- Design Overview
  - Summary
  - IOB Properties
  - Timing Constraints
  - Pinout Report
  - Clock Report
- Errors and Warnings
  - Synthesis Messages
  - Translation Messages
  - Map Messages
  - Place and Route Messages
  - Timing Messages
  - Bitgen Messages
  - All Current Messages
- Detailed Reports

Project Properties
- Enable Enhanced Design Summary
- Enable Message Filtering
- Display Incremental Messsages

Enhanced Design Summary Contents
- Show Errors
- Show Warnings
- Show Failing Constraints
- Show Clock Report

Design Summary

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Latches	104	9,600	1%	
Number of 4 input LUTs	301	9,600	3%	
Logic Distribution				
Number of occupied Slices	191	4,800	3%	
Number of Slices containing only related logic	191	191	100%	
Number of Slices containing unrelated logic	0	191	0%	
Total Number 4 input LUTs	309	9,600	3%	
Number used as logic	301			
Number used as a route-thru	8			
Number of bonded IOBs	134	406	33%	
Total equivalent gate count for design	2,374			
Additional JTAG gate count for IOBs	6,432			

Performance Summary			
Final Timing Score:	0	**Pinout Data:**	Pinout Report
Routing Results:	All Signals Completely Routed	**Clock Data:**	Clock Report
Timing Constraints:	All Constraints Met		

Detailed Reports					
Report Name	**Status**	**Generated**	**Errors**	**Warnings**	**Infos**
Synthesis Report	Current	Tue 20. Dec 11:53:26 2022	0	7 Warnings	5 Infos

Started : "Launching Design Summary".

Console | Errors | Warnings | Find in Files

Type here to search 29°C Partly sunny ENG 12:04 20-12-2022

CONCLUSION

In this proposed work, Truncated Approximate Carry based Booth Multiplier (TACBM) is presented with an error compensation circuit designed by selective modification of k-map to achieve twin goal of energy and error minimization. Extensive error analysis is performed by applying different parts of the compensation circuit to the non-truncated part. The design spaces of various multipliers are explored comprehensively. Further, addition of 4:2 and 5:2 compressors helps in reducing the complexity in partial product reduction. TACBM over its range provides 5.19% to 34.46% energy savings with MRED varying from 0.01% to 0.19%.

References

[1] A. D. Booth, "A signed binary multiplication technique," Quart. J. Mech. Appl. Math., vol. 4, no. 2, pp. 236–240, 1951.

[2] C. S. Wallace, "A suggestion for a fast multiplier," IEEE Trans. Electron. Comput., vol. EC-13, no. 1, pp. 14–17, Feb. 1964.

[3] L. Dadda, "Some schemes for parallel multipliers," Alta Frequenza, vol. 34, no. 5, pp. 349–356, Mar. 1965.

[4] E. L. Braun, Digital Computer Design: Logic, Circuitry, and Synthesis. New York, NY, USA: Academic, 2014.

[5] C. R. Baugh and B. A. Wooley, "A two's complement parallel array multiplication algorithm," IEEE Trans. Comput., vol. C-100, no. 12, pp. 1045–1047, Dec. 1973.

[6] D. Hampel, K. E. McGuire, and K. J. Prost, "CMOS/SOS serial parallel multiplier," IEEE J. Solid-State Circuits, vol. SSC-10, no. 5, pp. 307–313, Oct. 1975.

[7] Z. Huang and M. D. Ercegovac, "High-performance low-power leftto-right array multiplier design," IEEE Trans. Comput., vol. 54, no. 3, pp. 272–283, Mar. 2005.

[8] J. Prummel et al., "A 10 mW Bluetooth low-energy transceiver with on-chip matching," IEEE J. Solid-State Circuits, vol. 50, no. 12, pp. 3077–3088, Dec. 2015.

[9] J. Fadavi-Ardekani, "M×N Booth encoded multiplier generator using optimized Wallace trees," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 1, no. 2, pp. 120–125, Jun. 1993.

[10] N. Itoh, Y. Naemura, H. Makino, Y. Nakase, T. Yoshihara, and Y. Horiba, "A 600-MHz 54×54-bit multiplier with rectangular-styled Wallace tree," IEEE J. Solid-State Circuits, vol. 36, no. 2, pp. 249–257, Feb. 2001.

[11] V. G. Oklobdzija, D. Villeger, and S. S. Liu, "A method for speed optimized partial product reduction and generation of fast parallel multipliers using an algorithmic approach," IEEE Trans. Comput., vol. 45, no. 3, pp. 294–306, Mar. 1996.

[12] P. F. Stelling, C. U. Martel, V. G. Oklobdzija, and R. Ravi, "Optimal circuits for parallel multipliers," IEEE Trans. Comput., vol. 47, no. 3, pp. 273–285, Mar. 1998.

[13] A. A. Farooqui and V. G. Oklobdzija, "General data-path organization of a MAC unit for VLSI implementation of DSP processors," in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), vol. 2, May/Jun. 1998, pp. 260–263.

[14] N. Petra, D. De Caro, V. Garofalo, E. Napoli, and A. G. M. Strollo, "Truncated binary multipliers with variable correction and minimum mean square error," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 57, no. 6, pp. 1312–1325, Jun. 2010.

[15] J.-Y. Kang and J.-L. Gaudiot, “A simple high-speed multiplier design,” IEEE Trans. Comput., vol. 55, no. 10, pp. 1253–1258, Oct. 2006.
[16] S.-R. Kuang, J.-P. Wang, and C.-Y. Guo, “Modified booth multipliers with a regular partial product array,” IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 56, no. 5, pp. 404–408, May 2009.
[17] W. Yan, M. D. Ercegovac, and H. Chen, “An energy-efficient multiplier with fully overlapped partial products reduction and final addition,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 63, no. 11, pp. 1954–1963, Nov. 2016.
[18] J. Mori et al., “A 10 ns 54×54 b parallel structured full array multiplier with 0.5 µm CMOS technology,” IEEE J. Solid-State Circuits, vol. 26, no. 4, pp. 600–606, Apr. 1991.
[19] N. Ohkubo et al., “A 4.4 ns CMOS 54×54-b multiplier using passtransistor multiplexer,” IEEE J. Solid-State Circuits, vol. 30, no. 3, pp. 251–257, Mar. 1995.
[20] C.-H. Chang, J. Gu, and M. Zhang, “Ultra low-voltage low-power CMOS 4-2 and 5-2 compressors for fast arithmetic circuits,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 51, no. 10, pp. 1985–1997, Oct. 2004.
[21] L.-D. Van and J.-H. Tu, “Power-efficient pipelined reconfigurable fixed width Baugh-Wooley multipliers,” IEEE Trans. Comput., vol. 58, no. 10, pp. 1346–1355, Oct. 2009.5.
[22] Rajagopal, R., Karthick, R., Meenalochini, P., & Kalaichelvi, T. (2023). Deep Convolutional Spiking Neural Network optimized with Arithmetic optimization algorithm for lung disease detection using chest X-ray images. Biomedical Signal Processing and Control, 79, 104197.
[23] Srinivas, T. Aditya Sai, B. Ravindra Babu, Miskir Solomon Tsige, R. Rajagopal, S. Devi, and Subrata Chowdhury. "Effective implementation of the Prototype of a digital stethoscope using a Smartphone." In 2022 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES), pp. 1-8. IEEE, 2022.
[24] Rajagopal, R., M. Karthik, M. Soni, Narayan Krishan Vyas, S. Hemavathi, and M. R. Arun. "Monitoring the high-speed engine application using ferro magnetic system." Materials Today: Proceedings (2022).
[25] Babu, P. Ramesh, Vemuri Kusuma Priya, N. Drawin, R. Thiagarajan, and R. Krishnamoorthy. "Enhanced Hybrid Resource Scheduler for an Institution Employing NB-RR Scheduling." In 2022 8th International Conference on Smart Structures and Systems (ICSSS), pp. 1-6. IEEE, 2022.
[26] Jose, S. Edwin, R. Lal Raja Singh, and R. Rajagopal. "Automatic and real time classification of power quality disturbance using statistical moments." AIP Conference Proceedings. Vol. 2327. No. 1. AIP Publishing LLC, 2021.
[27] Rajagopal, R., and S. Edwin Jose. "An Efficient Framework for Locating Stroke in Brain MRI Images Using Radon Transform and Convolutional Neural Networks." In Next Generation of Internet of Things, pp. 385-395. Springer, Singapore, 2021.
[28] Rajagopal, R., and P. Subbaiah. "A survey on liver tumor detection and segmentation methods." ARPN Journal of Engineering and Applied Sciences 10, no. 6 (2015): 2681-2685
[29 Rajagopal, R., & Subbiah, P. (2014). Computer aided detection of liver tumor using SVM classifier. International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, 3(6), 10170-7.
[30] Rajagopal, R. (2019). Glioma brain tumor detection and segmentation using weighting random forest classifier with optimized ant colony features. International Journal of imaging systems and technology, 29(3), 353-359.